Protecting and Evaluating Genomic Privacy in Medical Tests and Personalized Medicine

Erman Ayday Jean Louis Raisaro
Jean-Pierre Hubaux
Laboratory for Communications and Applications
EPFL, Lausanne, Switzerland
firstname.lastname@epfl.ch

Jacques Rougemont
Bioinformatics and Biostatistics Core Facility
School of Life Sciences
EPFL, Lausanne, Switzerland
jacques.rougemont@epfl.ch

ABSTRACT

In this paper, we propose privacy-enhancing technologies for medical tests and personalized medicine methods that use patients' genomic data. Focusing on genetic disease-susceptibility tests, we develop a new architecture (between the patient and the medical unit) and propose a "privacy-preserving disease susceptibility test" (PDS) by using homomorphic encryption and proxy re-encryption. Assuming the whole genome sequencing to be done by a certified institution, we propose to store patients' genomic data encrypted by their public keys at a "storage and processing unit" (SPU). Our proposed solution lets the medical unit retrieve the encrypted genomic data from the SPU and process it for medical tests and personalized medicine methods, while preserving the privacy of patients' genomic data. We also quantify the genomic privacy of a patient (from the medical unit's point of view) and show how a patient's genomic privacy decreases with the genetic tests he undergoes due to (i) the nature of the genetic test, and (ii) the characteristics of the genomic data. Furthermore, we show how basic policies and obfuscation methods help to keep the genomic privacy of a patient at a high level. We also implement and show, via a complexity analysis, the practicality of PDS.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; J.3 [**Life and Medical Sciences**]: *Biology and genetics*; K.4.1 [**Computer and Society**]: Public Policy Issues—*Privacy*

Keywords

Genomic Privacy; Personalized Medicine; DNA; Security

1. INTRODUCTION

As a result of the rapid evolution in genomic research, substantial progress is expected in terms of improved diagnosis and better preventive medicine. However, the impact on privacy is unprecedented [7], because (i) the genome carries information about one's genetic conditions and predisposition to specific diseases (such as Alzheimer's) (ii) when someone voluntarily publishes his genome, important information about that person's relatives can be leaked (possibly against their will), (iii) complex privacy issues would arise if DNA analyses were to be used for both criminal investigations and healthcare purposes, and (iv) the extent of information that might be extracted by bioinformatics methods in the future cannot be foreseen.

Even though at this stage, the field of genomics is generally free of serious attacks, it is likely that the above threats will become more serious as the number of "sequenced" individuals becomes larger. Therefore, the need to adapt *privacy-enhancing technologies* (PETs) [16] to personal genomic data will continue to grow with time, as they are key tools for preventing an adversary from linking particular genomic data to a specific person or from inferring privacy-sensitive genomic data about a person.

Currently, the companies and hospitals that perform DNA sequencing store the genomic data of their customers and patients. Of course, tight legislation regulates their activities, but it is extremely difficult for them to protect themselves against the misdeeds of a hacker or a disgruntled employee. Therefore, new architectures and protocols are needed to store and process this privacy-sensitive genomic data, while still enabling its utilization by the medical units.

In this work, our goal is to protect the privacy of users' genomic data while enabling medical units to access the genomic data in order to conduct medical tests or develop personalized medicine methods. In a medical test, a medical unit checks for different health risks (e.g., disease susceptibilities) of a user by using specific parts of his genome. Similarly, to provide personalized medicine, a pharmaceutical company tests the compatibility of a user with a particular medicine. It is important to note that these genetic tests are currently done by different types of medical units, and the tools we propose in this paper aim to protect the genomic privacy of the patients in such tests. In both medical tests and personalized medicine methods, in order to preserve his privacy, the user does not want to reveal his complete genome to the medical unit or to the pharmaceutical company. In addition, in some scenarios, it is the pharmaceutical companies who do not want to reveal the genetic properties of their drugs. To achieve these goals, we introduce the *privacy-preserving disease susceptibility test* (PDS).

WPES'13, November 4, 2013, Berlin, Germany.
Copyright 2013 ACM 978-1-4503-2485-4/13/11 ...$15.00.
http://dx.doi.org/10.1145/2517840.2517843.

The main contributions of our work are summarized in the following.

1) We develop a new architecture between the patient and the medical unit. In particular, we propose to store the genomic data of the patients at a *storage and processing unit* (SPU) and conduct the computations on genomic data by using homomorphic encryption and proxy re-encryption to preserve the privacy of the genomic data.

2) The proposed PDS preserves the genomic privacy of a patient from a curious party at the SPU (who tries to infer the contents of the patient's DNA from his stored data). Furthermore, PDS ensures that a medical unit conducting genetic tests on a patient can only access the parts of the patient's DNA for which it is authorized.

3) The proposed PDS protects the privacy of a medical unit for the genetic properties of conducted genetic tests (especially, when the medical unit is a pharmaceutical company who does not want to reveal the genetic properties of its future drugs).

4) Using a real DNA profile, real genetic tests, and real values for the characteristics of the genomic data, we quantify a patient's genomic privacy (from a medical unit's point of view) as a result of the genetic tests he engages in. Moreover, we show how simple policies and obfuscation techniques help to reduce the decrease in the genomic privacy of the patient, while they still preserve the accuracy of the genetic tests.

5) We implement the proposed PDS and show its practicality via a complexity analysis.

The rest of the paper is organized as follows. In the next section, we summarize the related work on genomic privacy. In Section 3, we describe our proposed scheme for privacy-preserving medical tests and personalized medicine. In Section 4, we quantify the genomic privacy of a patient due to the genetic tests he undergoes, and we propose simple policies and obfuscation methods to minimize the privacy loss due to the genetic tests. In Section 5, we discuss the implementation of the proposed scheme and present its complexity and security evaluations. Finally, in Section 6, we conclude the paper.

2. RELATED WORK

We can put the research on the privacy of genomic data in three main categories: (i) private string searching and comparison, (ii) private release of aggregate data, and (iii) private clinical genomics.

In [32], Troncoso-Pastoriza *et al.* propose a protocol for string searching on DNA, which is then re-visited by Blanton and Aliasgari [12]. To compute the similarity of DNA sequences, in [23], Jha *et al.* propose using garbled circuits. In [14], Bruekers *et al.* propose privacy-enhanced comparison of DNA profiles by using homomorphic encryption. In their seminal work [26], Kantarcioglu *et al.* propose using homomorphic encryption to perform scientific investigations on integrated genomic data. In their scheme, all genomic data is encrypted by the same public key of the data storage site, and there is a single key holder site that can decrypt everything. As opposed to [26], we focus on the personal use of genomic data (e.g., in medical tests and personalized medicine methods) and we propose a method in which each user's genomic data is encrypted via his own cryptographic key. In one of the recent works [10], Baldi *et al.* make use of both medical and cryptographic tools for privacy-preserving paternity tests, personalized medicine, and genetic compatibility tests. Instead of utilizing public key encryption protocols, in [15], Canim *et al.* propose securing the biomedical data by using cryptographic hardware. Furthermore, Ayday *et al.* propose techniques for privacy-preserving use of genomic and non-genomic data in disease risk tests [9].

When releasing databases consisting of aggregate genomic data, it is shown that known privacy-preserving approaches (e.g., de-identification) are ineffective on (un-encrypted) genomic data [21,22,28,33,35]. Recently, using differential privacy was proposed by Fienberg *et al.* [20]; they aim to ensure that two aggregated genomic databases, differing from each other by only one individual's data, have indistinguishable statistical features.

In [17], utilizing a public cloud, Chen *et al.* propose a secure and efficient algorithm to align short DNA sequences to a reference (human) DNA sequence. Furthermore, in [34], Wang *et al.* propose a privacy-protection framework for important classes of genomic computations (e.g., search for homologous genes). Finally, Ayday *et al.* propose techniques for privacy-preserving management of raw genomes [8].

As a result of our extensive collaboration with geneticists, clinicians, and biologists, we concluded that a DNA string comparison is insufficient in many medical tests (that use genomic data) [5, 27, 30] and would not be enough to pave the way to personalized medicine. As it will become clearer in the next sections, for each genetic test, specific variants must be considered individually. Thus, as opposed to the aforementioned techniques, we use the individual variants of the users to conduct genetic tests. Furthermore, we quantify the genomic privacy of the users as a result of the genetic tests they undergo considering the statistical relationship between the variants, and we show techniques for keeping users' genomic privacy at high levels.

3. PETS FOR MEDICAL TESTS AND PERSONALIZED MEDICINE METHODS

Most medical tests and personalized medicine methods (that use genomic data) involve a patient and a medical unit. In general, the medical unit can be a physician in a medical center (e.g., hospital), a pharmacist, a pharmaceutical company, or a medical council. In this study, we consider the existence of a malicious entity in the medical unit as the potential attacker. That is, a medical unit might contain a disgruntled employee or it can be hacked by an intruder that is trying to obtain private genomic information about a patient (for which it is not authorized).

In addition, extreme precaution is needed for the storage of genomic data due to its sensitivity. Thus, we believe that a storage and processing unit (SPU) should be used to store the genomic data. We assume that the SPU is more "security-aware" than a medical unit, hence it can protect the stored genomic data against a hacker better than a medical unit (yet, attacks against the SPU cannot be ruled out, as we discuss next). Recent medical data breaches from various medical units [1] also support this assumption. Furthermore, instead of every medical unit individually storing the genomic data of the patients (in which case patients need to be sequenced by several medical units and their genomic data will be stored at several locations), a medical unit can retrieve the required genomic data belonging to a patient directly from the SPU. We note that a private company (e.g., cloud storage service), the government, or a non-profit organization could play the role of the SPU.

We assume that the SPU is an honest organization, but it might be curious. In other words, the SPU honestly follows the protocols and provides correct information to the other parties, however, a curious party at the SPU could access or infer the stored genomic data. Further, it is possible to identify a person only from his genomic data via phenotyping, which determines the observable physical or biochemical characteristics of an organism from its genetic makeup and environmental influences. Therefore, genomic data should be stored at the SPU in encrypted form. Similarly, apart from the possibility of containing a malicious entity, the medical unit honestly follows the protocols. Thus, we assume that the medical unit does not make malicious requests from the SPU.[1] We consider the following models for the attacker:

- A curious party at the SPU (or a hacker who breaks into the SPU), who tries to infer the genomic sequence of a patient from his stored genomic data. Such an attacker can infer the variants (i.e., nucleotides that vary between individuals) of the patient from his stored data.
- A malicious entity in the medical unit, who can be considered either as an attacker that hacks into the medical unit's system or a disgruntled employee who has access the medical unit's database. The goal of such an attacker is to obtain private genomic data of a patient for which it is not authorized. As we will discuss in Section 4, the main resource of such an attacker is the results of the genetic tests the patient undergoes.

We note that the attacker can also be a hacker who breaks into both the SPU and the medical unit at the same time (i.e., an attacker in the form of both above models). We will discuss this case in Section 3.4, and propose a solution against it.

3.1 Genomics Background

The human genome is encoded in double stranded DNA molecules consisting of two complementary polymer chains. Each chain consists of simple units called nucleotides (A,C,G,T). The human genome consists of approximately three billion letters. Even though more than 99% of these letters are identical in any two individual, there are differences between us due to genetic variations. Single nucleotide polymorphism (SNP) is the most common DNA variation in human population. A SNP is a position in the genome holding a nucleotide, which varies between individuals. For example, in Fig. 1, two sequenced DNA fragments from two different individuals contain a single different nucleotide at a particular SNP position (i.e., locus). Recent discoveries show that the susceptibility of a patient for several diseases can be computed from his SNPs [2, 25].

In general, there are two types of alleles (nucleotides) observed at a given SNP position: (i) the major allele is the most frequently observed nucleotide, and (ii) the minor allele is the rare nucleotide. Thus, each SNP is assigned a minor and a major allele frequency, among which the minor allele frequency is the lesser of the two. For instance, the two alleles for the SNP position in Fig. 1 are C and T (G and A in the figure are the complementary nucleotides for C and T, respectively).

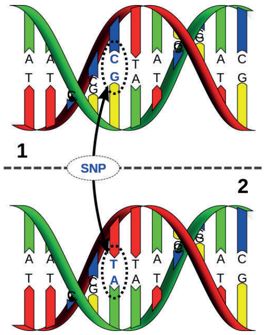
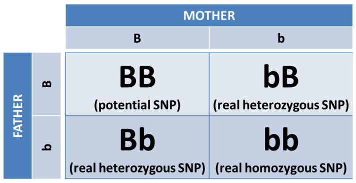


Figure 1: Single nucleotide polymorphism (SNP) with alleles C and T (© David Hall, License: Creative Commons).

Each SNP position includes two alleles (i.e., two nucleotides) and everyone inherits one allele of every SNP position from each of his parents. If an individual receives the same allele from both parents, he is said to be *homozygous* for that SNP position. If, however, he inherits a different allele from each parent (one minor and one major), he is called *heterozygous*. It is important to note that a SNP becomes a variant when it carries at least one minor allele. From here on, to avoid confusion, for each patient, we refer to these variants (i.e., SNPs carrying at least one minor allele) as the *real SNPs* and the remaining non-variants (i.e., SNPs carrying no minor alleles) as the *potential SNPs* of the patient; when we only say "SNPs", we mean both the real and potential SNPs. For example, assume that b is the minor allele and B is the major allele for a SNP position (both b and B are from the set $\{A, T, C, G\}$). Then, in Fig. 2, we illustrate the probable states of this SNP for an offspring, for different combinations of the father's and mother's alleles. We note that if the state of a SNP is known (potential, real homozygous or real heterozygous), its content (i.e., nucleotides residing at the corresponding SNP position) can be easily determined. Hence, hereafter, when we refer to the state of a SNP, we also mean its content.



		MOTHER	
		B	b
FATHER	B	BB (potential SNP)	bB (real heterozygous SNP)
	b	Bb (real heterozygous SNP)	bb (real homozygous SNP)

Figure 2: Probable states of a SNP for an offspring, given different combinations of his parents' alleles for the same SNP position.

There are approximately 50 million SNPs in the human population as of now (according to the NCBI dbSNP [3]) and each patient carries on average 4 million variants (i.e., real SNPs carrying at least one minor allele) out of this 50 million. We note that the number of SNPs in human population is increasing very rapidly [3], whereas the number of real SNPs per patient (around 4 million) remains the same. Moreover, this set of 4 million real SNPs is different for each patient.

3.2 Overview of the Proposed Scheme

For the simplicity of presentation, in the rest of this section, we will focus on a particular medical test (namely, computing genetic disease susceptibility). Similar techniques would apply for other medical tests and personalized medicine methods. In a typical genetic disease-susceptibility test, a *medical center* (MC) wants to check the susceptibility

[1] Even if malicious requests by the MC were possible, inference from such requests could be limited by using the proposed obfuscation technique in Section 4.2.

of a patient (P) for a particular disease X (i.e., the probability that patient P will develop disease X) by analyzing particular SNPs of the patient.[2]

For each patient, we propose to store only the real SNPs at the SPU. At this point, it can be argued that these 4 million real SNPs (nucleotides) could be easily stored on the patient's computer or mobile device, instead of at the SPU. However, we assert that this should be avoided due to the following issues. On one hand, types of variations in human population are not limited to SNPs, and there are other types of variations such as *copy-number variations* (CNVs), rearrangements, or translocations, consequently the required storage per patient is likely to be considerably more than only 4 million nucleotides. This high storage cost might still be affordable (via desktop computers or USB drives), however, the genomic data of the patient should be available any time (e.g., for emergencies), thus it should be stored at a reliable source such as the SPU. On the other hand, leaving the patient's genomic data in his own hands and letting him store it on his computer or mobile device is risky, because his mobile device can be stolen or his computer can be hacked. It is true that the patient's cryptographic keys (or his authentication material) to access his genomic data at the SPU can also be stolen, however, in the case of a stolen cryptographic key, his genomic data (which is stored at the SPU) will still be safe. This can be considered like a stolen credit card issue. If the patient does not report that his keys are compromised as soon as possible, his genomic data can be accessed by the attacker.

It is important to note that protecting only the states (contents) of the patient's real SNPs is not sufficient in terms of his genomic privacy. As the real SNPs are stored at the SPU, a curious party at the SPU can infer the nucleotides corresponding to the real SNPs from their positions and from the correlation between the patient's potential SNPs and the real ones (as discussed in Section 4). That is, by knowing the positions of the patient's real SNPs, the curious party at the SPU will at least know that the patient has one or two minor alleles at these SNP positions (i.e., it will know that the corresponding SNP position includes either a real homozygous or heterozygous SNP), and it can make its inference stronger using the correlation between the SNPs. Therefore, we propose to encrypt both the positions of the real SNPs and their states. We assume that the patient stores his cryptographic keys (public-secret key pair for asymmetric encryption, and symmetric keys between the patient and other parties) on his smart card (e.g., digital ID card). Alternatively, these keys can be stored at a cloud-based password manager and retrieved by the patient when required.

In short, the whole genome sequencing is done by a *certified institution* (CI) with the consent of the patient. Moreover, the real SNPs of the patient and their positions on the DNA sequence (or their unique IDs) are encrypted by the same CI (using the patient's public and symmetric key, respectively) and uploaded to the SPU, so that the SPU cannot access the real SNPs of the patient (or their positions). We are aware that the number of discovered SNPs increases with time. Thus, the patient's complete DNA sequence is also encrypted as a single vector file (via symmetric encryption using the patient's symmetric key) and stored at the SPU, thus when new SNPs are discovered, these can be included in the pool of the previously stored SNPs of the patient. We also assume the SPU not to have access to the real identities of the patients and data to be stored at the SPU by using pseudonyms; this way, the SPU cannot associate the conducted genetic tests to the real identities of the patients.

Depending on the access rights of the MC, either (i) the MC computes $\Pr(X)$, the probability that the patient will develop disease X by checking a subset of the patient's encrypted SNPs via homomorphic encryption techniques [13], or (ii) the SPU provides the relevant SNPs to the MC (e.g., for complex diseases that cannot be interpreted using homomorphic operations). These access rights are defined either jointly by the MC and the patient, or directly by the medical authorities. We note that homomorphic encryption lets the MC compute $\Pr(X)$ using encrypted SNPs of patient P. In other words, the MC does not access P's SNPs to compute his disease susceptibility. We use a modification of the Paillier cryptosystem [6,13] (described in Section 3.3) to support the homomorphic operations at the MC.

We describe the proposed *privacy-preserving disease susceptibility test* (PDS) in detail in Section 3.4. We also discuss the computation of genetic disease susceptibility by using homomorphic operations in Section 3.5.

3.3 Paillier Cryptosystem

Here, we briefly review the modified Paillier cryptosystem (described in detail in [6, 13]), which we use in this work, and its homomorphic properties.

The public key of patient P is represented as $(n, g, h = g^x)$, where the strong secret key is the factorization of $n = pq$ (p, q are safe primes), the secret key is $x \in [1, n^2/2]$, and g of order $(p-1)(q-1)/2$. Such a g can be easily found by selecting a random $a \in \mathbb{Z}^*_{n^2}$ and computing $g = -a^{2n}$.

Encryption of a message: To encrypt a message $m \in \mathbb{Z}_n$, we first select a random $r \in [1, n/4]$ and generate the ciphertext pair (T_1, T_2) as below:

$$T_1 = g^r \bmod n^2 \quad \text{and} \quad T_2 = h^r(1 + mn) \bmod n^2. \qquad (1)$$

Re-encryption of a message: An encrypted message (T_1, T_2) can be re-encrypted under the same public key, by using a new random number $r_1 \in [1, n/4]$ as below:

$$\hat{T_1} = g^{r_1} T_1 \bmod n^2 \quad \text{and} \quad \hat{T_2} = h^{r_1} T_2 \bmod n^2. \qquad (2)$$

Decryption of a message: The message m can be recovered as follows:

$$m = \Lambda(T_2/T_1^x), \qquad (3)$$

where $\Lambda(u) = \frac{(u-1) \bmod n^2}{n}$, for all $u \in \{u < n^2 \mid u = 1 \bmod n\}$.

Homomorphic properties: The below-mentioned homomorphic properties are supported by the modified Paillier cryptosystem:

- The product of two ciphertexts decrypts to the sum of their corresponding plaintexts.
- An encrypted plaintext raised to a constant decrypts to the product of the plaintext and the constant.

[2] In this study, we only focus on the diseases which can be analyzed using the SNPs. We admit that there are also other diseases which depend on other forms of mutations or environmental factors.

Proxy re-encryption: The patient's secret key x can be randomly divided into two shares: $x^{(1)}$ and $x^{(2)}$ (such that $x = x^{(1)} + x^{(2)}$). Using the above modified Paillier cryptosystem, an encrypted message (T_1, T_2) (under the patient's public key) can be partially decrypted by using $x^{(1)}$ to generate the ciphertext pair $(\tilde{T_1}, \tilde{T_2})$ as below:

$$\tilde{T_1} = T_1 \quad \text{and} \quad \tilde{T_2} = T_2 / T_1^{x^{(1)}} \bmod n^2. \tag{4}$$

Now, $(\tilde{T_1}, \tilde{T_2})$ can be decrypted using $x^{(2)}$ to recover the original message.

3.4 Privacy-Preserving Disease Susceptibility Test (PDS)

We assume that the state of SNP_i (SNP which resides at position i on the DNA sequence) for patient P is represented as SNP_i^P. Further, $\text{SNP}_i^P = 0$, if P does not have a variant at this position, $\text{SNP}_i^P = 1$, if P has a real heterozygous SNP at this position, and $\text{SNP}_i^P = 2$, if P has a real homozygous SNP at this position. We let Υ_P be the set positions (on the DNA sequence) at which patient P has real SNPs (i.e., at which $\text{SNP}_i^P \in \{1, 2\}$). We also let Ω_P represent the set of potential SNP positions (at which $\text{SNP}_i^P = 0$).

As the MC does the computations on the encrypted SNPs of the patient, if the MC has the complete secret key (x) of the patient, it can access the states of the SNPs used to conduct the genetic test (even though the MC may not be authorized to access the states of individual SNPs). Therefore, following a proxy re-encryption protocol that is proposed for the modified Paillier cryptosystem [6][3], the patient's secret key x is randomly divided into two shares: $x^{(1)}$ and $x^{(2)}$ (such that $x = x^{(1)} + x^{(2)}$). $x^{(1)}$ is given to the SPU and $x^{(2)}$ is given to the MC.

However, in some environments, splitting the secret key of the patient, and distributing the shares of the secret key to the SPU and MC might not be acceptable when (i) it is likely that the SPU and MC will collaborate to retrieve the patient's secret key, or (ii) neither the SPU nor the MC are security-aware, hence they can be hacked by the same attacker. Therefore, for the sake of completeness, in the following, we present PDS with and without proxy re-encryption.

3.4.1 *PDS with Proxy Re-encryption*

In some applications, the patient's involvement in the genetic test is not desired (except for his consent to the test). Thus, in this section, we present PDS with proxy re-encryption, in which the patient is only involved during the first part of the disease susceptibility test. Below, we summarize the PDS with proxy re-encryption. These steps are illustrated in Fig. 3.

- **Step 0:** The cryptographic keys (public and secret keys) of each patient are generated and distributed to the patients during the initialization period (public keys of the patients are also distributed to the CI, the SPU, and to the MC). Then, symmetric keys are established between the parties (among which interaction occurs in Fig. 3). By using the symmetric keys, the positions of P's real SNPs are encrypted (as we discuss next) and the communication between the parties is protected from an eavesdropper. The distribution, update and revocation of cryptographic keys are handled by a trusted entity.

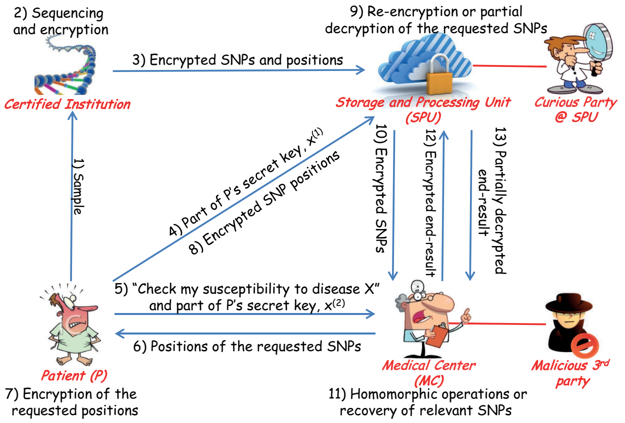

2) Sequencing and encryption
3) Encrypted SNPs and positions
9) Re-encryption or partial decryption of the requested SNPs
Certified Institution
Storage and Processing Unit (SPU)
Curious Party @ SPU
1) Sample
4) Part of P's secret key, x(1)
8) Encrypted SNP positions
10) Encrypted SNPs
12) Encrypted end-result
13) Partially decrypted end-result
5) "Check my susceptibility to disease X" and part of P's secret key, x(2)
6) Positions of the requested SNPs
Patient (P)
Medical Center (MC)
Malicious 3rd party
7) Encryption of the requested positions
11) Homomorphic operations or recovery of relevant SNPs

Figure 3: PDS with proxy re-encryption.

- **Step 1:** The patient (P) provides his sample (e.g., his saliva) to the certified institution (CI) for sequencing.

- **Step 2:** After sequencing, the CI first determines the positions of P's real SNPs and constructs Υ_P (set positions at which P has real SNPs).

After constructing Υ_P, the CI encrypts each element in Υ_P with the symmetric key k_{PC} (between patient P and the CI, established and distributed during Step 0) by using a secure symmetric encryption function and generates $\mathcal{E}(\Upsilon_P, k_{PC}) = \{\mathcal{E}(v_i, nonce_i, k_{PC}) : v_i \in \Upsilon_P\}$, where $nonce_i = \mathfrak{F}(k_{PC}, v_i)$, and $\mathfrak{F}$ is a pseudorandom function. The CI also encrypts the states (contents) of the real SNPs of the patient P using P's public key to obtain $\text{E}(\text{SNP}_i^P, g^x)$ and $\text{E}((\text{SNP}_i^P)^2, g^x)$ for each real SNP of P.[4] Furthermore, the CI encrypts a "0" value (representing the states of the potential SNPs with positions in Ω_P) using P's public key to obtain $\text{E}(0, g^x)$ and $\text{E}(0^2, g^x)$. Finally, the CI associates an arbitrary position, v_0, for this "0" value and encrypts v_0 using the symmetric key k_{PC} to obtain $\mathcal{E}(v_0, nonce_0, k_{PC})$.

- **Step 3:** The CI sends the encrypted SNPs and positions to the SPU. Eventually, for each real SNP of P, the SPU stores $[\mathcal{E}(v_i, nonce_i, k_{PC}) \mid \text{E}(\text{SNP}_i^P, g^x) \mid \text{E}((\text{SNP}_i^P)^2, g^x)]$ ($\text{SNP}_i^P \in \{1, 2\}$ as only the real SNPs of the patient are stored) and to represent the potential SNPs of P, the SPU stores only $[\mathcal{E}(v_0, nonce_0, k_{PC}) \mid \text{E}(0, g^x) \mid \text{E}(0^2, g^x)]$. We let the SPU know that the encrypted position $\mathcal{E}(v_0, nonce_0, k_{PC})$ represents the potential SNPs of P.

- **Step 4:** The patient provides a part of his secret key ($x^{(1)}$) to the SPU.

- **Step 5:** The MC wants to conduct a susceptibility test on P for a particular disease X, and P provides the other part of his secret key ($x^{(2)}$) to the MC. As discussed before, by distributing the parts of patient's secret key to the SPU and MC, we avoid exposing the patient's secret key (x) to the MC, thus we prevent the MC from accessing more information than it is authorized for.

[3] We follow the original works [6,13] for the notations of some cryptographic primitives (e.g., proxy re-encryption, secret key).

[4] The squared value of each real SNP (e.g., $(\text{SNP}_i^P)^2$) is also encrypted and stored in order to realize the homomorphic operations discussed in Section 3.5.

• **Step 6:** The MC tells the patient the positions of the SNPs that are required for the susceptibility test or requested directly as the relevant SNPs (but not the individual contributions of these SNPs to the test). The positions of the requested SNPs should be revealed by the MC to determine (i) whether the MC has sufficient access rights to operate on the requested SNPs of the patient, and (ii) whether the patient gives consent to the use of the requested SNPs by the MC. We note that when the medical unit is embodied in a pharmaceutical company (who does not want to reveal the genetic properties of its drugs), the pharmaceutical company might request additional (i.e., dummy) SNPs from the patient (along with the actual ones that are used in the test) in order to hide the precise nature of the request.

• **Step 7:** The patient encrypts each requested position υ_j with the symmetric key k_{PC} to obtain $\mathcal{E}(\upsilon_j, nonce_j, k_{PC})$. We note that this operation can be easily done via the patient's smart card (e.g., by scanning the card at the MC as a consent to the test).

• **Step 8:** The patient sends the SPU the encrypted positions of the requested SNPs.

• **Step 9:** The SPU receives each requested position, υ_j, from the patient in an encrypted form ($\mathcal{E}(\upsilon_j, nonce_j, k_{PC})$). If the patient has a real SNP at the requested position (i.e., $\mathcal{E}(\upsilon_j, nonce_j, k_{PC}) \in \mathcal{E}(\Upsilon_P, k_{PC})$), the SPU retrieves the encrypted SNP (i.e., the encrypted state of the corresponding SNP) at the corresponding (encrypted) location. Otherwise, if the patient has a potential SNP at the requested position (i.e., $\mathcal{E}(\upsilon_j, nonce_j, k_{PC}) \notin \mathcal{E}(\Upsilon_P, k_{PC})$), the SPU retrieves the encrypted SNP (i.e., encrypted state) stored at $\mathcal{E}(\upsilon_0, nonce_0, k_{PC})$. It is important to note that the SPU does not know the virtue of the genetic test for which the patient is being tested, hence it cannot launch an attack to infer the susceptibility of the patient for the corresponding disease by counting the number of requested positions at which the patient has a real SNP.

Then, one of the following two scenarios occur at the SPU:

(a) If the end-result is to be computed (by the MC), the retrieved SNPs are re-encrypted at the SPU under the patient's public key. Following the properties of the modified Paillier cryptosystem, an encrypted SNP (using a random $r \in [1, n/4]$) can be re-encrypted under the same public key, by using a new random number $r_1 \in [1, n/4]$ (re-encryption under the same public key is discussed in Section 3.3). As there is only one value stored at the SPU representing the states of the potential SNPs at which P does not have a variant (at position $\mathcal{E}(\upsilon_0, nonce_0, k_{PC})$), this value is re-encrypted for each different request of a non-variant, so that the MC cannot infer the positions of the non-variants of the patient.

(b) If relevant SNPs are requested (by the MC), the SPU partially decrypts the retrieved SNPs by using $x^{(1)}$ following a proxy re-encryption protocol (described in detail in Section 3.3).

• **Step 10:** Re-encrypted (or partially decrypted) SNPs are sent to the MC (by the SPU) in the same order as they are requested in Step 6.

• **Step 11:** One of the following occurs at the MC:

(a) If the end-result is to be computed (by the MC), the MC computes P's total susceptibility for disease X by using the homomorphic properties (i.e., homomorphic addition and multiplication with a constant) of the modified Paillier cryptosystem (as discussed in Section 3.5) under the patient's public key.

(b) If relevant SNPs are requested (by the MC), the MC decrypts the message received from the SPU by using $x^{(2)}$ and recovers the relevant SNPs.

The remaining steps are executed only if the end-result of the test is computed by the MC; if the relevant SNPs are requested, Step 11 is the last step of the protocol.

• **Step 12:** The MC sends the encrypted end-result to the SPU.

• **Step 13:** The SPU partially decrypts the end-result using $x^{(1)}$ by following a proxy re-encryption protocol and sends it back to the MC.

• **Step 14:** The MC decrypts the message received from the SPU by using $x^{(2)}$ and recovers the end-result.

3.4.2 PDS without Proxy Re-encryption

In this approach, the shares of P's secret key are not distributed to the SPU or MC. Most of this approach is the same as PDS with proxy re-encryption. The only difference (other than the fact that the parts of P's secret key are not distributed to the other parties) is the transfer of the end-result or the relevant SNPs to the MC as follows:

• If the relevant SNPs are requested by the MC, the SPU sends the encrypted SNPs (by P's public key) to P. P decrypts these SNPs (using his secret key) and sends them to the MC.

• If the end-result of the susceptibility test is requested by the MC, the disease-susceptibility test is done (via homomorphic operations) at the MC and the encrypted end-result is sent to P. Then, P decrypts the end-result and sends it back to the MC.

The above operations put more burden on the patient during the protocol. However, we emphasize that these operations can be smoothly done on the patient's smart card without requiring a substantial effort from the patient himself.

3.5 Computing Disease Susceptibility via Homomorphic Operations

The MC uses a proper function to compute P's predicted disease susceptibility via homomorphic encryption. There are different functions for computing the predicted susceptibility. In [27], Kathiresan *et al.* propose to count the number of unfavorable alleles carried by the patient for each SNP related to a particular disease. Similarly, in [5], Ashley *et al.* propose to multiply the likelihood ratios (LRs) of the most important SNPs for a particular disease. Furthermore, a *weighted averaging* function can also be used, which computes the predicted susceptibility by weighting the SNPs by their contributions. It is important to note that these functions are currently used by geneticists to check patients' susceptibilities for a wide variety of diseases. In the following, we discuss how to compute the predicted disease susceptibility at the MC by using weighted averaging (which is an advanced version of the function proposed in [27]) from the encrypted SNPs (the function proposed in [5] can be also computed similarly).

Again, we let SNP_i^P represent the state of SNP_i for patient P, where $\mathrm{SNP}_i^P \in \{0, 1, 2\}$. Assume that the susceptibility for disease X is determined by the set of SNPs, whose positions (on the DNA sequence) are in set φ_X.[5] The contributions of different states of SNP_i^P (for $i \in \varphi_X$) to the susceptibility for disease X are computed via previous studies (on case and control populations) and they are already known by the MC. That is, $\mathrm{p}_0^i(X) \triangleq \Pr(X|\mathrm{SNP}_i^P = 0)$, $\mathrm{p}_1^i(X) \triangleq \Pr(X|\mathrm{SNP}_i^P = 1)$, and $\mathrm{p}_2^i(X) \triangleq \Pr(X|\mathrm{SNP}_i^P = 2)$ ($i \in \varphi_X$) are determined and known by the MC. Further, the contribution of SNP_i to the susceptibility for disease X is denoted by C_i^X. Note that these contributions are also computed by previous studies on case and control groups and they are known by the MC.

As we have discussed before (in Section 3.4.1), the SPU stores the states of P's SNPs along with their squared values, encrypted by P's public key $(n, g, h = g^x)$. The squared values of the SNPs' states are stored in order to realize the genetic disease susceptibility test via homomorphic operations as discussed next. Thus, the MC uses $\mathrm{E}(\mathrm{SNP}_i^P, g^x)$ and $\mathrm{E}((\mathrm{SNP}_i^P)^2, g^x)$ ($i \in \varphi_X$) for the computation of predicted susceptibility of P for disease X. Furthermore, the MC uses the following data for the disease susceptibility test: (i) the markers for disease X (whose positions are in φ_X), (ii) corresponding probabilities (p_j^i(X), $i \in \varphi_X$ and $j \in \{0, 1, 2\}$), and (iii) the contributions of each SNP (C_i^X).

Next, the MC computes the predicted susceptibility of patient P for disease X, $\mathbb{S}_P^X$, by using weighted averaging. This can be computed in plaintext as below:

$$\mathbb{S}_P^X = \frac{1}{\sum\limits_{i\in\varphi_X} C_i^X} \sum_{i\in\varphi_X} C_i^X \times \mathrm{p}^i_{\mathrm{SNP}_i^P}(X) \times \mathrm{SNP}_i^P$$

$$= \frac{1}{\sum\limits_{i\in\varphi_X} C_i^X} \times \sum_{i\in\varphi_X} C_i^X \Bigg\{ \frac{\mathrm{p}_0^i(X)}{(0-1)(0-2)} \Big[\mathrm{SNP}_i^P - 1\Big] \times$$

$$\Big[\mathrm{SNP}_i^P - 2\Big] + \frac{\mathrm{p}_1^i(X)}{(1-0)(1-2)} \Big[\mathrm{SNP}_i^P - 0\Big] \Big[\mathrm{SNP}_i^P - 2\Big] +$$

$$\frac{\mathrm{p}_2^i(X)}{(2-0)(2-1)} \Big[\mathrm{SNP}_i^P - 0\Big] \Big[\mathrm{SNP}_i^P - 1\Big] \Bigg\}. \qquad (5)$$

In short, (5) takes all SNPs that are associated with disease X (in φ_X), determines the contribution of each SNP to the disease based on its state ($\{0, 1, 2\}$), multiplies this value with the general contribution of the SNP (C_i^X), and sums up for all SNPs in φ_X. The computation in (5) can be realized by using the encrypted SNPs of the patient (and by utilizing the homomorphic properties of the modified Paillier cryptosystem) to compute the encrypted susceptibility, $\mathrm{E}(\mathbb{S}_P^X, g^x)$ as below:

$$\mathrm{E}(\mathbb{S}_P^X, g^x) = \Bigg\{ \prod_{i\in\varphi_X} \Big\{ \Big[\mathrm{E}((\mathrm{SNP}_i^P)^2, g^x)\Big]^{(\tilde{a}+\tilde{b}+\tilde{c})}$$

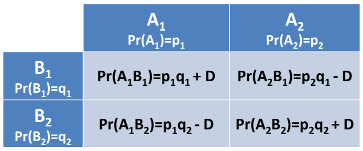
$$\times \Big[\mathrm{E}(\mathrm{SNP}_i^P, g^x)\Big]^{(-3\tilde{a}-2\tilde{b}-\tilde{c})} \times [\mathrm{E}(2\tilde{a}, g^x)] \Big\}^{C_i^X} \Bigg\}^{\frac{1}{\sum\limits_{i\in\varphi_X} C_i^X}}, \qquad (6)$$

where $\tilde{a} = \mathrm{p}_0^i(X)/2$, $\tilde{b} = -\mathrm{p}_1^i(X)$, and $\tilde{c} = \mathrm{p}_2^i(X)/2$. We note that $\mathrm{E}(2\tilde{a}, g^x)$ can be computed using P's public key, as $\tilde{a}$ is known by the MC. Furthermore, as the modified Paillier cryptosystem requires to work on integer values, for the implementation, the MC multiplies each non-integer value $\tilde{a}$, $\tilde{b}$, $\tilde{c}$, C_i^X, and $(\sum\limits_{i\in\varphi_X} C_i^X)^{-1}$ with a large constant (as will be discussed in Section 5.1). Next, the MC sends $\mathrm{E}(\mathbb{S}_P^X, g^x)$ to the SPU. Then, the SPU partially decrypts the end-result $\mathrm{E}(\mathbb{S}_P^X, g^x)$ using its share $(x^{(1)})$ of P's secret key (x), as discussed before, to obtain $\mathrm{E}(\mathbb{S}_P^X, g^{x^{(2)}})$ and sends it back to the MC. Finally, the MC decrypts $\mathrm{E}(\mathbb{S}_P^X, g^{x^{(2)}})$ using its share $(x^{(2)})$ of P's secret key to recover the end-result $\mathbb{S}_P^X$.

4. QUANTIFICATION OF GENOMIC PRIVACY AT THE MEDICAL CENTER

From the results of genetic tests, the MC can learn more information than it is authorized by using (i) the characteristics of the exposed SNPs, and (ii) disease markers and their contributions. Therefore, in this section, we will quantify the genomic privacy of the patient from the MC's point of view as a result of the genetic tests he encounters. We assume that the MC tries to infer the states of particular SNPs of the patient (regardless whether they are real or potential SNPs) from the results of genetic tests. As a result, the genomic privacy of the patient changes over time (as he has more genetic tests). Consequently, our goals are to (i) compute the decrease in the genomic privacy of patient P as a result of the genetic tests he undergoes, and (ii) show how simple policies and obfuscation methods help to keep the genomic privacy of the patient at high levels.

An important issue to consider in this evaluation is the *linkage disequilibrium* (LD) between SNPs [19]. LD occurs when SNPs at two loci (SNP positions) are not independent of each other. For example, assume that SNP_i and SNP_j (SNPs which reside at positions i and j on the DNA sequence, respectively) are in LD. Let (A_1, A_2) and (B_1, B_2) be the potential alleles for these two SNP positions (i.e., loci) i and j. Further, let (p_1, p_2) and (q_1, q_2) be the allele probabilities of (A_1, A_2) and (B_1, B_2), respectively. That is, the probability that an individual will have A_1 as the first allele[6] of SNP_i is p_1, and so on. If there were no LD (i.e., if SNP_i and SNP_j were independent), the probability that an individual will have both A_1 and B_1 as the first alleles of SNP_i and SNP_j would be p_1q_1. However, due to the LD, this probability is equal to $p_1q_1 + D$, where D represents the LD between these two SNP positions. In Fig. 4, we illustrate this LD relationship for all possible combinations of (A_1, A_2) and (B_1, B_2). We note that D can be either negative or positive, depending on the LD between two loci. The strength of the LD between two SNP positions is usually repre-



	A_1 $\Pr(A_1)=p_1$	A_2 $\Pr(A_2)=p_2$
B_1 $\Pr(B_1)=q_1$	$\Pr(A_1B_1)=p_1q_1 + D$	$\Pr(A_2B_1)=p_2q_1 - D$
B_2 $\Pr(B_2)=q_2$	$\Pr(A_1B_2)=p_1q_2 - D$	$\Pr(A_2B_2)=p_2q_2 + D$

Figure 4: Linkage disequilibrium (LD) between two SNP positions with potential alleles (A_1, A_2) and (B_1, B_2), respectively.

[5] SNPs whose positions are in φ_X are not necessarily among the real SNPs of patient P (i.e., P does not need to have a real homozygous or heterozygous SNP at those positions).

[6] As we discussed before, each SNP position includes two alleles (i.e., two nucleotides).

sented as $\mathrm{r}^2 = \frac{D^2}{p_1 p_2 q_1 q_2}$, where $\mathrm{r}^2 = 1$ represents the strongest LD relationship. In Fig. 5, we illustrate the r^2 values between a set of SNP pairs (which are used in this study).

As in Fig. 5, r^2 values are symmetric between the SNP pairs. However, when the LD is represented as a conditional probability (e.g., probability that the first allele of SNP_i is A_1, given the first allele of SNP_j is B_1), the probabilities become asymmetric between the SNP pairs. We computed the conditional probabilities between the SNP pairs (which are in LD) using their D and r^2 values from [24] and [11] (r^2 value, along with other genetic information, which is out of the scope of this paper, is required to determine the sign of D). For this study, we use these conditional probabilities for the inference of the MC.

As we have discussed before, the MC can obtain the result of a genetic test in two different ways:

Test 1: MC obtains a subset of P's SNPs (e.g., for complex diseases for which the disease susceptibility cannot be computed using homomorphic operations).

Test 2: MC obtains the end-result of a genetic test, which is conducted using homomorphic operations as discussed in Section 3.5. We assume that weighted averaging is used in Test 2, to conduct the disease susceptibility test at the MC (as described in Section 3.5).

In Test 2, even though the MC does not have access to the patient's SNPs, the following information is known by the MC and it helps the MC infer the patient's un-exposed SNPs more accurately: (i) the markers of diseases and their contributions (C_i^j values), (ii) the contributions of the different alleles of a SNP for a particular disease (i.e., contributions of different states of a SNP for a particular disease), and (iii) the LD values between the SNPs.

We use *asymmetric entropy* [29] to compute the genomic privacy of the patient (from the MC's view point) due to the genetic tests he undergoes. Asymmetric entropy is defined as

$$h(p_i) = \frac{p_i(1-p_i)}{(-2w+1)p_i + w^2}, \qquad (7)$$

where p_i is the probability of correctly inferring the state (content) of SNP_i for patient P (i.e., SNP_i^P), and w is the point at which the entropy is maximum. We do not use the traditional entropy metric [31] to quantify privacy, as the initial probability for correctly inferring the state of a SNP position (with no information about the patient) depends on known statistics, hence the maximum value of the entropy is different for each SNP position, while the range of p_i is always the same. Consider a SNP_i with minor allele probability of p_a and major allele probability of p_A. Then, (i) if $\mathrm{SNP}_i^P = 0$, $w = p_A^2$, (ii) if $\mathrm{SNP}_i^P = 1$, $w = 2p_A p_a$, and (iii) if $\mathrm{SNP}_i^P = 2$, $w = p_a^2$.

Assume that the set W includes the SNPs over which we compute the genomic privacy of patient P. Further, let $\mathrm{D_W}$ represent the set of diseases whose risk factors (i.e., patients' susceptibilities for these diseases) are computed using different subsets of SNPs in W. To compute the genomic privacy of patient P, we add the entropies of all of P's SNPs, over which we compute his genomic privacy as $\sum_{i \in \mathrm{W}} h(p_i)$. Patient P has the highest genomic privacy when the MC does not have any information about the patient's DNA sequence, hence the entropies of his SNPs are at maximum (from the MC's point of view). As the patient undergoes Test 1 or Test 2, his genomic privacy starts decreasing due to his revealed or inferred SNPs.

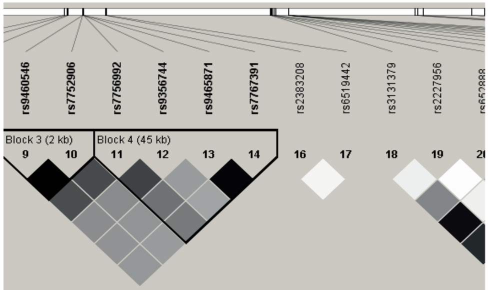


Figure 5: r^2 values, which are a measure of the LD strength between two SNP positions, between a set of SNP pairs (which are used in this study) using the Haploview software [11]. Darker squares represent higher r^2 values, hence stronger LD between the SNPs.

For the simplicity of the presentation, we consider slotted time and we assume that, at each time slot, the MC conducts a test (either Test 1 or Test 2) to the patient. For Test 1, we let the number of exposed SNPs be integer values, randomly chosen from the set $[10, 15]$. Once the SNPs are revealed, the MC updates its inference on the non-exposed SNPs by using the LD relationships between the exposed SNPs and those non-exposed. For Test 2, the MC conducts the disease susceptibility test on patient P for a disease in $\mathrm{D_W}$ and obtains only the end-result of the test. However, the MC can compute the potential end-results of such a test using the information it possesses (markers, contributions, allele probabilities, etc.) for every possible combination of the SNPs for that disease. Considering that the patient is being tested for disease D_i and the markers (SNPs) revealing D_i are from the set φ_i, there are $3^{|\varphi_i|}$ potential values for the end-result of the susceptibility test (since each SNP can be in 3 different states). If only one of these potential end-results matches the actual end-result of the test, the MC automatically learns the states (contents) of all the SNPs of P in φ_i. If however, multiple potential end-results match the actual end-result, the inferred probabilities on the SNPs are updated proportionally to the occurrence probabilities of the matching potential end-results. To save time, the MC can compute the potential end-results of the susceptibility tests offline. As a result of Test 2, if any SNP is completely revealed, its LD relationships with the non-exposed SNPs are also used for further inference (of the non-exposed SNPs).

For the evaluation of privacy loss on a real example, we constructed W from the markers of 40 diseases in $\mathrm{D_W}$ (such as Alzheimer's, Parkinson's, etc.) from [25] and [2]. Eventually, we computed the genomic privacy of the patient using around 500 SNPs (among which the patient has both potential and real SNPs). For the patient's DNA profile, we used a real human DNA profile from [4]. As we discussed before, we computed the conditional probabilities due to the LD between the SNPs (in W) from [24] and [11]. We note that each disease in $\mathrm{D_W}$ has at least 1, and at most 15 markers. Further, we observed that there are 12 SNPs that are the markers of more than one disease.

In Fig. 6, we illustrate the decrease in the genomic privacy of patient P during 40 occurrences of a random combination

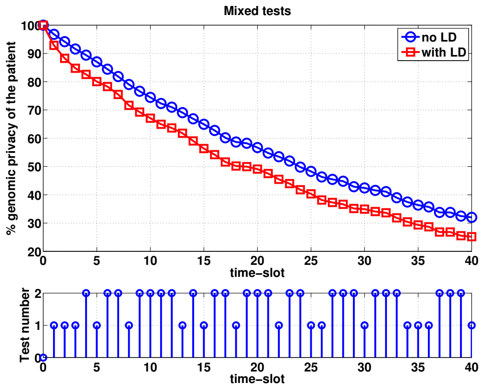


Figure 6: Genomic privacy of patient P (computed using (7)) from MC's point of view during 40 occurrences of a random combination of Test 1 and Test 2, when (i) the LD relationships between the SNPs are ignored, and (ii) the LD relationships are also used for the inference.

of Test 1 and Test 2. To observe the effect of the LD to the inference of the SNPs in W, we illustrate the decrease in the genomic privacy of the patient both by ignoring and considering the LD between the SNPs in W. As we illustrate in the figure, genomic privacy drops rapidly, especially when the LD values are utilized (we observed about 8% more decrease in the genomic privacy when the LD values are utilized), hence after each test, the MC infers increasingly more about the states (contents) of the SNPs in W. It is important to note that the research on the LD relationships between the SNPs is still ongoing, and as the field of genomics becomes more mature, we expect stronger inference of the SNPs due to the LD relationships. Due to this rapid decrease in the genomic privacy, we argue that either some policies should be enforced to the MC (for Test 1) or some obfuscation methods should be used at the SPU before providing the end-result to the MC (for Test 2). In the following subsections, we discuss some basic potential policies and obfuscation methods and show how such basic techniques can help to keep the genomic privacy of the patient at high levels. We emphasize that the purpose of the next two subsections is not to introduce novel policies and obfuscation methods; it is to show how simple precautions would help to reduce the decrease in the genomic privacy of the patient.

4.1 Enforcing Policies

Here, we propose a basic policy on the MC, in which the MC has to delete the exposed SNPs (as a result of Test 1) from its database after a definite time. Let Σ be the set of SNPs that are exposed to the MC as a result of Test 1 at time t. With this proposed policy, the MC can keep these SNPs in its database only until time $t+\tau$. That is, an attacker (who hacks into the MC's database) or a disgruntled employee, who happens to access MC's database sometime between time t and $t+\tau$, can use the SNPs in Σ to infer other non-exposed SNPs (via the LD relationships) of the patient. However, an attacker or disgruntled employee who accesses to MC's database after time $t+\tau$ cannot see the states of the SNPs in Σ, thus cannot use them to infer other SNPs.

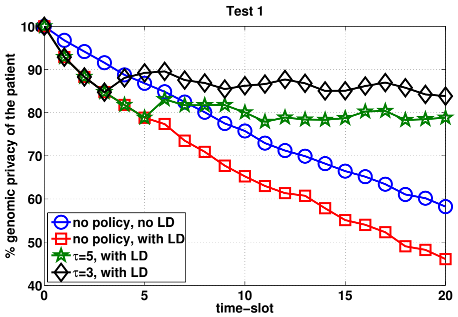


Figure 7: Genomic privacy of the patient (computed using (7)) from MC's point of view during 20 occurrences of Test 1, when (i) there is no policy and the LD relationships between the SNPs are ignored, (ii) there is no policy and the LD relationships are also used for the inference, (iii) $\tau = 5$ while considering the LD relationships, and (iv) $\tau = 3$ while considering the LD relationships.

In short, even though the MC is not a malicious unit itself, deleting the exposed SNPs from MC's database after a definite time would allow less information to be revealed to a potential attacker. In Fig. 7, we show the change in the genomic privacy of the patient for 20 occurrences of Test 1 when there is no policy (first ignoring and then considering the LD to show the effect of the LD) and for two different values of τ ($\tau = 3$ and 5 while considering the LD). We observe that the patient can re-gain his lost genomic privacy, up to some level, when his exposed SNPs (in Test 1) are deleted from the MC's database. As opposed to Test 1, we do not foresee that the MC would delete the results of Test 2 from its database because, Test 2 directly reports the susceptibility of the patient for a particular disease, hence the MC would keep this result in the medical file of the patient for future use.

4.2 Obfuscation Methods

For most genetic tests (e.g., genetic disease susceptibility tests), it is sufficient (for the MC) to obtain the range that the end-result falls in. Thus, as a simple obfuscation method, we propose to provide the end-result of Test 2 to the MC as a *range*. However, as the end-result of Test 2 is encrypted at the SPU, SPU cannot apply an obfuscation method to the end-result itself. Thus, for the practicality of this obfuscation method, we propose either (i) sending the encrypted end-result (from the SPU) to another entity called *obfuscation unit* (OU) (which does not know patient's real identity or the nature of the genetic test), or (ii) using a privacy-preserving comparison protocol [18] between the SPU and the MC (to compare two encrypted values).

In the former approach, we assume that the OU possesses the same part of P's secret key as the MC ($x^{(2)}$). The OU decrypts the end-result using $x^{(2)}$, applies the obfuscation method, re-encrypts the obfuscated result using patient's public key (g^x), and sends it back to the SPU. Then, once again the SPU partially decrypts the obfuscated end-result by using its share of patient's secret key ($x^{(1)}$) and sends it to the MC.[7] We also assume that the OU and the MC

[7] In the case of PDS without proxy re-encryption, obfuscation can be handled by the patient's smart card, as the

do not collude. In the latter approach, through a message exchange with the MC, the SPU compares the (encrypted) end-result of the genetic test with some pre-defined boundaries to determine the range that the end-result falls in (refer to [18] for details). It is important to note that during this process, neither the SPU nor the MC learns the end-result of the genetic test or the outcome of the comparison protocol. Once the SPU obtains the encrypted result of the comparison protocol, it can compute the (encrypted) range that the end-result of the genetic test falls in, and send this encrypted range to the MC. In the following, we briefly discuss this simple obfuscation method and its impact on the patient's genomic privacy.

Let the entire result range, $[0, 1]$, be divided into a smaller ranges of equal size. Then, the SPU determines (via the OU, or a private comparison protocol) which range the end-result of the test maps to and reports the corresponding range to the MC. For example, assume that the entire result range is divided into $a = 4$ smaller ranges $[0, 0.25)$, $[0.25, 0.50)$, $[0.50, 0.75)$, and $[0.75, 1]$. Then, if the end-result of the genetic test is 0.3241, MC will only learn that the end-result is in the range $[0.25, 0.50)$. If a private comparison protocol [18] is preferred for obfuscation, the SPU asks the MC for $a - 1$ comparisons (i.e., to compare the encrypted end-result with $a - 1$ pre-defined boundaries). In this case, as a increases, the number of required comparisons also increases (we discuss the complexity of the private comparison protocol in Section 5.1). In Fig. 8, we illustrate the change in the genomic privacy of the patient for 20 occurrences of Test 2 and for different values of a. Obviously, as a decreases, the utility (i.e., accuracy of the genetic test result) at the MC also decreases, but the genomic privacy of the patient decreases slower. We note that there is no universal value for a in today's genetic tests, and the optimal value of a (for maximum utility) might change based on the type of the test. For example, a recent study on cardiovascular disease shows that $a = 4$ is sufficient to determine a patient's susceptibility to this disease [30]. Comparing Figs. 7 and 8, we also notice that the contribution of the LD to the inference is higher for Test 1.

Finally, in Fig. 9, we show the decrease in the genomic privacy of the patient during 40 occurrences of a random combination of Test 1 and Test 2, for different policy and obfuscation parameters (τ and a). We observe that we can obtain significantly higher genomic privacy by using both policies and obfuscation methods at the same time (with $\tau = 3$ and $a = 10$) compared to the case with no precautions.

5. EVALUATION AND IMPLEMENTATION OF PDS

5.1 Implementation and Complexity Evaluation

To evaluate the practicality of the proposed privacy-preserving algorithm, we implemented it and assessed its storage requirement and computational complexity on Intel Core i7-2620M CPU with 2.70 GHz processor under Windows 7. We computed the disease susceptibility using weighted averaging (at the MC, see Section 3.5) and a real DNA profile from [4]. Further, we computed the disease susceptibility for real diseases using their corresponding markers from [25] and [2].[8] Our implementation is in Java and it relies on the MySQL 5.5 database.

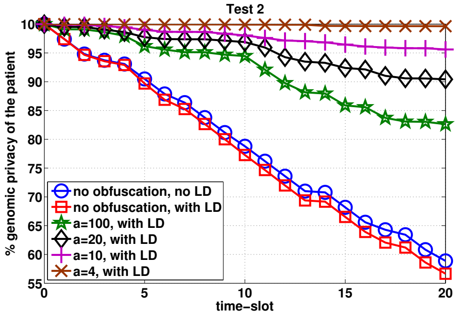


Figure 8: Genomic privacy of the patient (computed using (7)) from MC's point of view during 20 occurrences of Test 2, when (i) there is no obfuscation and the LD relationships between the SNPs are ignored, (ii) there is no obfuscation and the LD relationships are also used for the inference, (iii) the entire result range, $[0, 1]$, is divided into a smaller ranges of equal size ($a = 4, 10, 20, 100$), while considering the LD relationships.

In Table I, we summarize the computational and storage complexities of the proposed method at (i) certified institution (CI), (ii) SPU, (iii) MC, and (iv) P, with and without proxy re-encryption, and when the size of the security parameter (n in the modified Paillier cryptosystem) is set to 2048 (2K) and 4096 (4K) bits. We evaluate the proposed methods considering the following costs: (i) encryption of patient's real SNPs, (ii) disease-susceptibility test at the MC via homomorphic operations (using 10 SNPs), (iii) decryption of the end-result (or relevant SNPs), (iv) proxy re-encryption, (v) re-encryption under the same public key, and (vi) storage cost.[9] We used the CCM mode of AES to encrypt the positions of the SNPs (Step 2 in Section 3.4.1). We do not illustrate the computational cost of this operation in Table I as it is negligible compared to Paillier encryption/decryption and homomorphic operations. Finally, if a private comparison protocol [18] is preferred for obfuscation, a single comparison takes around 450 ms. when $n = 2$K, and 1 sec. when $n = 4$K.

We emphasize that the encryption of the SNPs at the CI is a one-time operation and is significantly faster than the sequencing and analysis of the sequence (which takes days). Further, this encryption (along with the re-encryption under the same public key) can be conducted much more efficiently by computing some parameters, such as (g^r, h^r) pairs, offline for various r values, for each patient. Indeed, by computing (g^r, h^r) pairs offline, we observe that the encryption takes only 0.049 ms per SNP (when $n = 2$K) and 0.168 ms per SNP (when $n = 4$K) at the CI. Similarly, using offline computations, re-encryption under the same public key takes 0.182 ms per SNP (when $n = 2$K) and 0.658 ms per SNP

patient already decrypts the end-result before sending it to the MC.

[8] As the Paillier cryptosystem requires to work on integer values, we multiplied each SNP contribution and allele probability by 10^3.

[9] We conducted each operation in Table I for at least 1000 times and got an average.

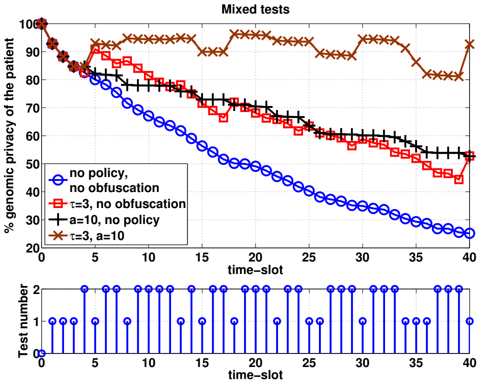
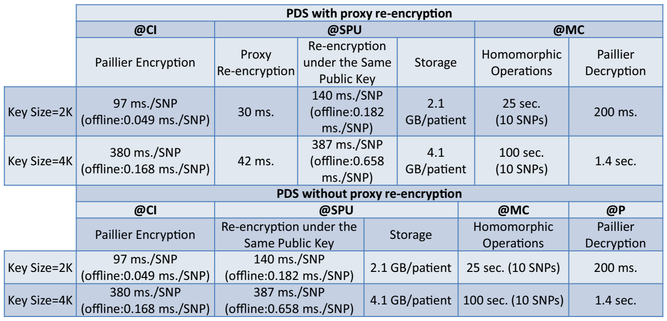


Figure 9: Genomic privacy of the patient (computed using (7)) from MC's point of view during 40 occurrences of a random combination of Test 1 and Test 2, when (i) there is no precautions and the LD relationships are used for the inference, and (ii) different policy and obfuscation parameters (τ and a) are applied, while considering the LD relationships.



	PDS with proxy re-encryption					
	@CI	@SPU			@MC	
	Paillier Encryption	Proxy Re-encryption	Re-encryption under the Same Public Key	Storage	Homomorphic Operations	Paillier Decryption
Key Size=2K	97 ms./SNP (offline:0.049 ms./SNP)	30 ms.	140 ms./SNP (offline:0.182 ms./SNP)	2.1 GB/patient	25 sec. (10 SNPs)	200 ms.
Key Size=4K	380 ms./SNP (offline:0.168 ms./SNP)	42 ms.	387 ms./SNP (offline:0.658 ms./SNP)	4.1 GB/patient	100 sec. (10 SNPs)	1.4 sec.

	PDS without proxy re-encryption				
	@CI	@SPU		@MC	@P
	Paillier Encryption	Re-encryption under the Same Public Key	Storage	Homomorphic Operations	Paillier Decryption
Key Size=2K	97 ms./SNP (offline:0.049 ms./SNP)	140 ms./SNP (offline:0.182 ms./SNP)	2.1 GB/patient	25 sec. (10 SNPs)	200 ms.
Key Size=4K	380 ms./SNP (offline:0.168 ms./SNP)	387 ms./SNP (offline:0.658 ms./SNP)	4.1 GB/patient	100 sec. (10 SNPs)	1.4 sec.

Table 1: Computational and Storage Complexities of the Proposed Method.

(when $n = 4$K) at the SPU. In summary, all these numbers show the practicality of our privacy-preserving algorithm.

5.2 Security Evaluation

The proposed scheme preserves the privacy of patients' genomic data relying on the security strength of the modified Paillier cryptosystem. The extensive security evaluation of the modified Paillier cryptosystem can be found in [13]. Below we summarize two important security features of this cryptosystem.

- One-wayness: No efficient adversary has any significant chance of finding a pre-image to the ciphertext when he sees only the ciphertext and the public key of the patient. It is shown in [13] that the one-wayness of the modified Paillier cryptosystem can be related to the *lift Diffie-Hellman* problem which is shown to be as hard as the partial *discrete logarithm* problem.
- Semantic security: An adversary will be unable to distinguish pairs of ciphertexts based on the message they encrypt. It is shown in [13] that if *decisional Diffie-Hellman assumption* in $\mathbb{Z}^*_{n^2}$ holds, then the modified Paillier cryptosystem is semantically secure.

If the secret key of the patient, x, is randomly divided and distributed to the storage and processing unit (SPU) and medical center (MC) as in Section 3.4.1, this secret key could be revealed if the MC colludes with the SPU (or the same attacker hacks into both the SPU and the MC), but the factors n, p, and q remain secret. In Section 3.4.2, we present an alternative approach that avoids the distribution of the patient's secret key to other parties and is thus robust against such a collusion.

Overall, the proposed scheme (with or without proxy re-encryption) provides a high level of privacy for the patients' genomic data because, from the view point of a curious party at the SPU, inferring the states of the patient's real SNPs (i.e., homozygous or heterozygous) with the stored information is equivalent to inferring them with no information about the patient. Furthermore, as the positions of the real SNPs are also encrypted, a curious party at the SPU cannot infer the states of the real SNPs from their positions. Another advantage of PDS is that individual contributions of the genetic variant markers remain secret at the MC, because the homomorphic operations are conducted at the MC. This advantage could become more significant when this approach is used for personalized medicine methods in which a pharmaceutical company (embodied in this case as the medical unit) does not want to reveal the genetic properties (i.e., marker contributions) of its drugs.

6. CONCLUSION

In this paper, we have introduced a privacy-preserving scheme for the utilization of genomic data in medical tests and personalized medicine methods. We have shown that the encrypted genomic data of the patient can be stored at a storage and processing unit (SPU) and processed at the medical unit using homomorphic encryption and proxy re-encryption while preserving the patient's privacy. Moreover, we have quantified the genomic privacy of the patient based on the genetic tests he undergoes and showed how simple policies and obfuscation methods help to reduce the decrease in the genomic privacy of the patient. We are confident that our proposed privacy-preserving scheme will encourage the use of genomic data by both individuals and medical units.

7. REFERENCES

[1] http://articles.washingtonpost.com/2012-06-02/national/35462326_1_data-breaches-medical-data-social-security-numbers.

[2] http://www.eupedia.com/genetics/medical_dna_test.shtml.

[3] http://www.ncbi.nlm.nih.gov/projects/SNP/.

[4] http://www.ncbi.nlm.nih.gov/projects/SNP/snp_ind.cgi?ind_id=10.

[5] E. Ashley, A. Butte, M. Wheeler, R.Chen, and T. Klein. Clinical assessment incorporating a personal genome. *The Lancet*, 375(9725):1525–1535, 2010.

[6] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security*, 9:1–30, Feb. 2006.

[7] E. Ayday, E. D. Cristofaro, G. Tsudik, and J. P. Hubaux. The chills and thrills of whole genome sequencing. *arXiv:1306.1264*, 2013.

[8] E. Ayday, J. L. Raisaro, U. Hengartner, A. Molyneaux, and J. P. Hubaux. Privacy-preserving processing of raw genomic data. *Proceedings of DPM International Workshop on Data Privacy Management*, 2013.

[9] E. Ayday, J. L. Raisaro, P. J. McLaren, J. Fellay, and J. P. Hubaux. Privacy-preserving computation of disease risk by using genomic, clinical, and environmental data. *Proceedings of USENIX Security Workshop on Health Information Technologies*, 2013.

[10] P. Baldi, R. Baronio, E. De Cristofaro, P. Gasti, and G. Tsudik. Countering GATTACA: Efficient and secure testing of fully-sequenced human genomes. *Proceedings of the 18th ACM Conference on Computer and Communications Security*, 2011.

[11] J. Barrett, B. Fry, J. Maller, and M. Daly. Haploview: Analysis and visualization of LD and haplotype maps. *Bioinformatics 21*, 2005.

[12] M. Blanton and M. Aliasgari. Secure outsourcing of DNA searching via finite automata. *Proceedings of the 24th Annual IFIP WG 11.3 Working Conference on Data and Applications Security and Privacy*, pages 49–64, 2010.

[13] E. Bresson, D. Catalano, and D. Pointcheval. A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications. *Proceedings of Asiacrypt*, 2003.

[14] F. Bruekers, S. Katzenbeisser, K. Kursawe, and P. Tuyls. Privacy-preserving matching of DNA profiles. Technical report, 2008.

[15] M. Canim, M. Kantarcioglu, and B. Malin. Secure management of biomedical data with cryptographic hardware. *IEEE Transactions on Information Technology in Biomedicine*, 16(1), 2012.

[16] A. Cavoukian. Privacy by design. 2009. `http://www.ontla.on.ca/library/repository/mon/23002/289982.pdf`.

[17] Y. Chen, B. Peng, X. Wang, and H. Tang. Large-scale privacy-preserving mapping of human genomic sequences on hybrid clouds. *Proceeding of the 19th Network and Distributed System Security Symposium*, 2012.

[18] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft. Privacy-preserving face recognition. *Proceedings of the 9th International Symposium on Privacy Enhancing Technologies*, pages 235–253, 2009.

[19] D. S. Falconer and T. F. Mackay. *Introduction to Quantitative Genetics (4th Edition)*. Addison Wesley Longman, Harlow, Essex, UK, 1996.

[20] S. E. Fienberg, A. Slavkovic, and C. Uhler. Privacy preserving GWAS data sharing. *Proceedings of the IEEE 11th International Conference on Data Mining Workshops*, Dec. 2011.

[21] M. Gymrek, A. L. McGuire, D. Golan, E. Halperin, and Y. Erlich. Identifying personal genomes by surname inference. *Science: 339 (6117)*, pages 321–324, Jan. 2013.

[22] N. Homer, S. Szelinger, M. Redman, D. Duggan, and W. Tembe. Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. *PLoS Genetics*, 4, Aug. 2008.

[23] S. Jha, L. Kruger, and V. Shmatikov. Towards practical privacy for genomic computation. *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, pages 216–230, 2008.

[24] A. Johnson, R. Handsaker, S. Pulit, M. Nizzari, C. O'Donnell, and P. de Bakker. SNAP: A web-based tool for identification and annotation of proxy SNPs using HapMap. *Bioinformatics 24(24):2938-2939*, 2008.

[25] A. D. Johnson and C. J. O'Donnell. An open access database of genome-wide association results. *BMC Medical Genetics 10:6*, 2009.

[26] M. Kantarcioglu, W. Jiang, Y. Liu, and B. Malin. A cryptographic approach to securely share and query genomic sequences. *IEEE Transactions on Information Technology in Biomedicine*, 12(5):606–617, 2008.

[27] S. Kathiresan, O. Melander, D. Anevski, C. Guiducci, and N. Burtt. Polymorphisms associated with cholesterol and risk of cardiovascular events. *The New England Journal of Medicine*, 358:1240–1249, 2008.

[28] B. Malin and L. Sweeney. How (not) to protect genomic data privacy in a distributed network: Using trail re-identification to evaluate and design anonymity protection systems. *Journal of Biomedical Informatics*, 37:179–192, Jun. 2004.

[29] S. Marcellin, D. Zighed, and G. Ritschard. An asymmetric entropy measure for decision trees. *Proceedings of International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 1292–1299, 2006.

[30] M. Rotger and *et al.* Contribution of genetic background, traditional risk factors and HIV-related factors to coronary artery disease events in HIV-positive persons. *Clinical Infectious Diseases*, Mar. 2013.

[31] A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. *Proceedings of Privacy Enhancing Technologies Symposium*, 2002.

[32] J. R. Troncoso-Pastoriza, S. Katzenbeisser, and M. Celik. Privacy preserving error resilient DNA searching through oblivious automata. *Proceedings of the 14th ACM Conference on Computer and Communications Security*, pages 519–528, 2007.

[33] R. Wang, Y. F. Li, X. Wang, H. Tang, and X. Zhou. Learning your identity and disease from research papers: Information leaks in genome wide association study. *Proceedings of the 16th ACM Conference on Computer and Communications Security*, pages 534–544, 2009.

[34] R. Wang, X. Wang, Z. Li, H. Tang, M. K. Reiter, and Z. Dong. Privacy-preserving genomic computation through program specialization. *Proceedings of the 16th ACM Conference on Computer and Communications Security*, pages 338–347, 2009.

[35] X. Zhou, B. Peng, Y. F. Li, Y. Chen, H. Tang, and X. Wang. To release or not to release: Evaluating information leaks in aggregate human-genome data. *Proceedings of the 16th European Conference on Research in Computer Security*, 2011.